## AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions of claims in the application:

**Listing of Claims:**

1. (Currently Amended) A computer implemented executable code check system comprising:

an input component that receives an object file and a specification associated with the object file, the specification comprising information associated with a plug-in condition for a method; and,

a checker that employs the specification to facilitate static checking of the object file, the checker passing a user injected custom state to the plug-in condition to check a fault condition and providing information if the fault condition is determined, the checker performing a component-wise comparison of the user injected custom state and a state defined by a parameter to determine the fault condition.

2. (Original) The system of claim 1, the plug-in condition comprising a precondition for the method.

3. (Original) The system of claim 2, the checker providing information associated with an object's state after a call to the method, the information being based, at least in part, upon the plug-in precondition.

4. (Original) The system of claim 1, the plug-in condition comprising a postcondition for the method.

5. (Original) The system of claim 4, the checker providing information associated with an object's state after a call to the method, the information being based, at least in part, upon the plug-in postcondition.

6. (Original) The system of claim 1, the object file being based, at least in part, upon a language that compiles to Common Language Runtime.

7. (Original)The system of claim 1, the object file being based, at least in part, upon at least one of C#, Visual Basic.net and Managed C++.

8. (Original) The system of claim 1, the specification comprising information associated with a state-machine protocol.

9. (Original) The system of claim 8, a state of an object modeled with a custom state.

10. (Original) The system of claim 9, the state of the object further being modeled with a custom state component.

11. (Original) The system of claim 10, the specification comprising at least one of a plug-in precondition and a plug-in postcondition method, which is a method of the custom state that is invoked by the checker to perform interface-specific state checks and state transitions.

12. (Original) The system of claim 1, wherein the specification is embedded with the object file.

13. (Original) The system of claim 1, wherein the specification is stored in a specification repository.

14. (Original) The system of claim 1, further comprising a specification extractor that queries a database for its schema and stores information associated with the schema in a specification repository.

15. (Currently Amended) A method of facilitating static checking of executable code comprising:

receiving executable code;

receiving a specification associated with the executable code, the specification comprising information associated with at least one of a precondition [[and]] or a postcondition for a method;

statically applying the specification to the executable code by passing a user injected custom state to the at least one precondition or postcondition;

performing a component-wise comparison of the user injected custom state and a state defined by a parameter to determine a fault condition;

determining whether [[a]] the fault condition exists based, at least in part, upon the statically applied specification; and,

providing information associated with the fault condition, if a fault condition is determined to exist.

16. (Original) A computer readable medium having stored thereon computer executable instructions for carrying out the method of claim 15.

17. (Currently Amended) A method of developing a software component comprising:

implementing a subclass of a custom state class;

implementing at least one of a plug-in precondition or a plug-in postcondition as a method of the subclass;

placing a custom attribute on an enclosing type declaration that references the custom state subclass;

placing an attribute on a declaration that references the at least one of a plug-in precondition or a plug-in postcondition;

performing a component-wise comparison of a user injected custom state and a state defined by a parameter to determine a fault condition; and

determining [[a]] the fault condition based in part upon the information from the at least one of plug-in precondition or a plug-in postcondition.

18. (Original) A computer readable medium having stored thereon computer executable instructions for carrying out the method of claim 17.

19. (Currently Amended) A method of performing static checking of executable code comprising:

invoking a precondition plug-in, providing the precondition plug-in with a program execution state;

receiving information from the precondition plug-in;

performing a component-wise comparison of a user injected custom state and a state defined by a parameter to determine a fault condition;

determining whether [[a]] the fault condition exists based, at least in part, upon the information from the pre-condition plug-in; and,

providing information associated with the fault condition, if a fault condition is determined to exist.

20. (Original) The method of claim 19, further comprising at least one of the following:

invoking a postcondition plug-in, providing the postcondition plug-in with the program execution state; and,

receiving information from the postcondition plug-in.

21. (Cancelled)

22. (Currently Amended) A computer readable medium storing computer executable components of an executable code check system comprising:

an input component that receives an object file and a specification associated with the object file, the specification comprising information associated with a plug-in condition for a method; and,

a checker component that employs the specification to facilitate static checking of the object file, the checker component passing a user injected custom state to the plug-in condition to check a fault condition and providing information if the fault condition is determined, the

checker performing a component-wise comparison of the user injected custom state and a state defined by a parameter to determine the fault condition.

23. (Previously Presented) A computer implemented executable code check system comprising:

means for receiving a specification associated with an object file, the specification comprising information associated with a plug-in condition for a method;

means for statically checking the object file based, at least in part, upon the specification;

means for passing a user injected custom state to the plug-in condition and determining if a fault condition exists;

means for performing a component-wise comparison of the user injected custom state and a state defined by a parameter to determine the fault condition; and,

means for providing information if a fault condition is determined to exist.

24. (Previously Presented) A method of performing static checking of executable code comprising:

receiving a request, the request including a parameter;

setting a type of a result of a method call to a type of the parameter;

employing the parameter only during static checking of the method; and

performing component-wise comparison of a user injected custom state and a state defined by the parameter to determine a fault condition.